
# ADSA MiniProblem Report

Amir Mahmoudi
Yassine Lahbabi

December 2020

# Contents

# 1 Introduction

Welcome to our Advanced Data Structure and Algorithm. This project needed to be done in a group of 2. We've done all this project in python, the purpose of this report is to explain our methodology, choices and discuss our results. Have a nice reading !

# 2 Step 1 : Organize the tournament

## 2.1 Data Structure

### 2.1.1 Player

For the player we've use a class data structure.A player is defined by is name/id and his score (the score is initiated at None) hence when we create a player if we only specify is name, the score will be None. The player represent a node in our score data structure, you will see it in next section.
The other information regarding a player are :

- His score, if it's not mention when the player is created with attribute the value zero in the other case we keep the value. It's an int value.
- His life status, it's a Boolean initiated at True meaning he's alive.
- His role, the variable is self.impostor, it's a Boolean initiated at False meaning he's a crew mate.When the game starts this variable can change to True implies he's now an impostor.
- His children, the player is a node in our score data structure so we mention if he has a left or right child they are both Boolean initiated at False because the Score data structure is not created.

Other than the init function we have the str function return a string to describe the player for instance we can have the string :

Amir is alive, and he is an Impostor

### 2.1.2 Score

To stock the score of all players and have a leader board we decided to use an AVL Tree for various reason, first, the insertion and deletion was simple to implement we have also the possibility to re balance our AVL Tree. But why re balance is essential to this structure ? You will see this part in the In Order Traversal paragraph

**Insertion**
For insertion, if the AVL tree is empty we initiate the player inserted as the root. If the AVL isn't empty we will insert the player node by comparing his score to the AVL's nodes. After inserting the player we re balance our tree to keep it balanced.

**Deletion**
For deletion, After deleting the player we re balance our tree to keep it balance.

**Re Balance**

**In Order Traversal**
For In Order Traversal, we use a recursive method we are going through all the tree adding to a list, the left sub tree of a node, the node in question and the last the right sub tree of this node. Let's manually compute the function with this example.

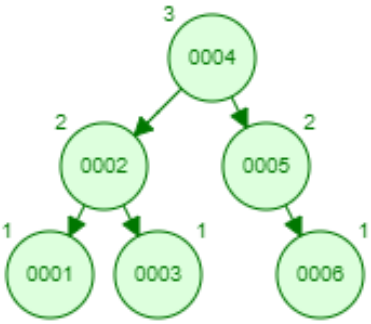


Figure 1: AVL Tree

First the algorithm will check the node 0004 but it has children so we check first his left child that's node 0002. This node has children so we check his left child that's node 0001. This node doesn't have children so we add it to our list, we add then the node 0002 we check is right child 0003 same reason for node 0001. We came back to node 0004 we add it to the list.
So far our list is [1,2,3,4]. We now at node 0004 and we need to check his right child, so we are now at node 0005, this node doesn't have a left child so we append him to our list and then check his right child. We're finally at node 0006, we've checked all the nodes and our list is : [1,2,3,4,5,6].
To conclude, the In Order Traversal is useful the obtain directly the leader board.

### 2.1.3 Tournament

## 2.2 Methodology

## 3 Step 2 : You>Guybrush Threepwood>Professor Layton

# 4 Step 3 : I don't see him, but I can give proofs he vents !

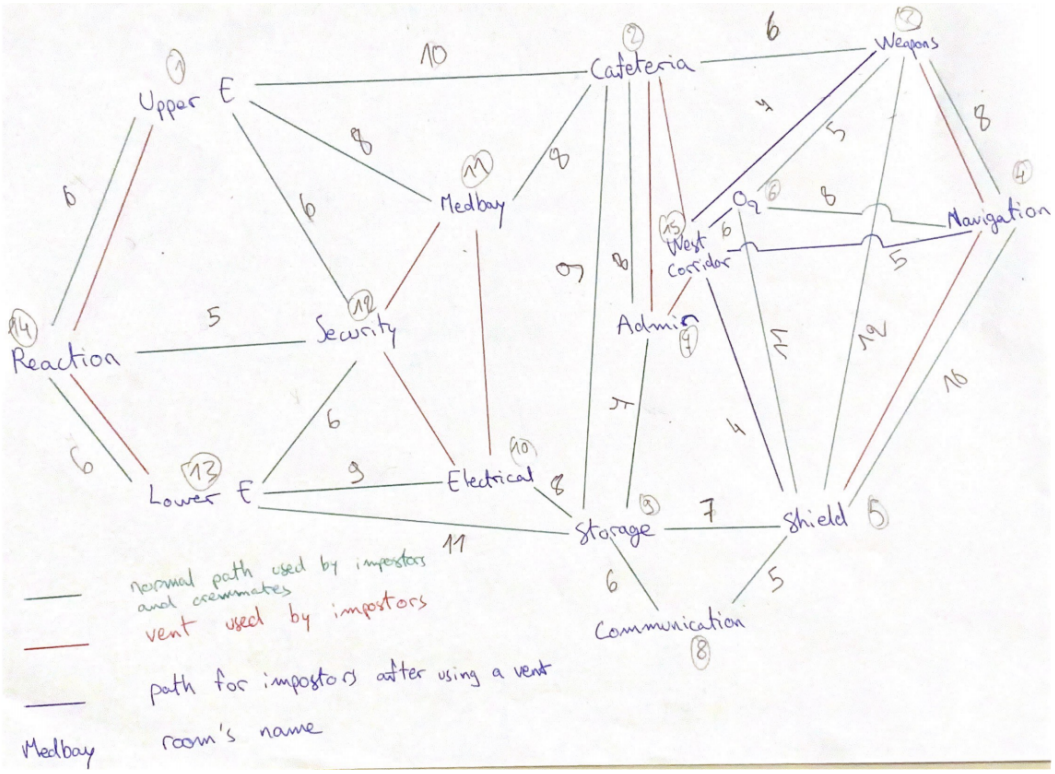


Figure 2: Weighted Graph

We need two models for the graph because the impostors can use an extra vertex : West Corridor and also we need to change the weight value, for instance going from Upper E to Reaction will take 6 seconds for a crew mate but an impostor can use the vent so we have a weight of zero.Hence, we need to implement a graph for the impostors and another for the crew mates.
We obtained the weights in figure right above by measuring with a ruler the distance between each rooms. Each centimeter is a weight of 1
Argue pathfinding
By running the code we have the time travel between a pair of rooms

| | Upper E | Cafeteria | Weapons | Navigations | Shield | O2 | Admin | Communication | Storage | Electrical | MedBay | Security | Lower E | Reactor | West Corridor |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Upper E | 0.0 | 10.0 | 14.0 | 14.0 | 14.0 | 16.0 | 10.0 | 17.0 | 11.0 | 5.0 | 5.0 | 5.0 | 0.0 | 0.0 | 10.0 |
| Cafeteria | 10.0 | 0.0 | 4.0 | 4.0 | 4.0 | 6.0 | 0.0 | 9.0 | 7.0 | 8.0 | 8.0 | 8.0 | 10.0 | 10.0 | 0.0 |
| Weapons | 14.0 | 4.0 | 0.0 | 0.0 | 0.0 | 5.0 | 4.0 | 5.0 | 7.0 | 12.0 | 12.0 | 12.0 | 14.0 | 14.0 | 4.0 |
| Navigations | 14.0 | 4.0 | 0.0 | 0.0 | 0.0 | 5.0 | 4.0 | 5.0 | 7.0 | 12.0 | 12.0 | 12.0 | 14.0 | 14.0 | 4.0 |
| Shield | 14.0 | 4.0 | 0.0 | 0.0 | 0.0 | 5.0 | 4.0 | 5.0 | 7.0 | 12.0 | 12.0 | 12.0 | 14.0 | 14.0 | 4.0 |
| O2 | 16.0 | 6.0 | 5.0 | 5.0 | 5.0 | 0.0 | 6.0 | 10.0 | 12.0 | 14.0 | 14.0 | 14.0 | 16.0 | 16.0 | 6.0 |
| Admin | 10.0 | 0.0 | 4.0 | 4.0 | 4.0 | 6.0 | 0.0 | 9.0 | 7.0 | 8.0 | 8.0 | 8.0 | 10.0 | 10.0 | 0.0 |
| Communication | 17.0 | 9.0 | 5.0 | 5.0 | 5.0 | 10.0 | 9.0 | 0.0 | 6.0 | 14.0 | 14.0 | 14.0 | 17.0 | 17.0 | 9.0 |
| Storage | 11.0 | 7.0 | 7.0 | 7.0 | 7.0 | 12.0 | 7.0 | 6.0 | 0.0 | 8.0 | 8.0 | 8.0 | 11.0 | 11.0 | 7.0 |
| Electrical | 5.0 | 8.0 | 12.0 | 12.0 | 12.0 | 14.0 | 8.0 | 14.0 | 8.0 | 0.0 | 0.0 | 0.0 | 5.0 | 5.0 | 8.0 |
| MedBay | 5.0 | 8.0 | 12.0 | 12.0 | 12.0 | 14.0 | 8.0 | 14.0 | 8.0 | 0.0 | 0.0 | 0.0 | 5.0 | 5.0 | 8.0 |
| Security | 5.0 | 8.0 | 12.0 | 12.0 | 12.0 | 14.0 | 8.0 | 14.0 | 8.0 | 0.0 | 0.0 | 0.0 | 5.0 | 5.0 | 8.0 |
| Lower E | 0.0 | 10.0 | 14.0 | 14.0 | 14.0 | 16.0 | 10.0 | 17.0 | 11.0 | 5.0 | 5.0 | 5.0 | 0.0 | 0.0 | 10.0 |
| Reactor | 0.0 | 10.0 | 14.0 | 14.0 | 14.0 | 16.0 | 10.0 | 17.0 | 11.0 | 5.0 | 5.0 | 5.0 | 0.0 | 0.0 | 10.0 |
| West Corridor | 10.0 | 0.0 | 4.0 | 4.0 | 4.0 | 6.0 | 0.0 | 9.0 | 7.0 | 8.0 | 8.0 | 8.0 | 10.0 | 10.0 | 0.0 |

Figure 3: Floyd Warshall Matrix for Impostors

| | Upper E | Cafeteria | Weapons | Navigations | Shield | O2 | Admin | Communication | Storage | Electrical | MedBay | Security | Lower E | Reactor |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Upper E | 0.0 | 10.0 | 16.0 | 24.0 | 26.0 | 21.0 | 18.0 | 25.0 | 19.0 | 21.0 | 8.0 | 6.0 | 12.0 | 6.0 |
| Cafeteria | 10.0 | 0.0 | 6.0 | 14.0 | 16.0 | 11.0 | 8.0 | 15.0 | 9.0 | 17.0 | 8.0 | 16.0 | 20.0 | 16.0 |
| Weapons | 16.0 | 6.0 | 0.0 | 8.0 | 12.0 | 5.0 | 14.0 | 17.0 | 15.0 | 23.0 | 14.0 | 22.0 | 26.0 | 22.0 |
| Navigations | 24.0 | 14.0 | 8.0 | 0.0 | 10.0 | 8.0 | 22.0 | 15.0 | 17.0 | 25.0 | 22.0 | 30.0 | 28.0 | 30.0 |
| Shield | 26.0 | 16.0 | 12.0 | 10.0 | 0.0 | 11.0 | 14.0 | 5.0 | 7.0 | 15.0 | 24.0 | 24.0 | 18.0 | 24.0 |
| O2 | 21.0 | 11.0 | 5.0 | 8.0 | 11.0 | 0.0 | 19.0 | 16.0 | 18.0 | 26.0 | 19.0 | 27.0 | 29.0 | 27.0 |
| Admin | 18.0 | 8.0 | 14.0 | 22.0 | 14.0 | 19.0 | 0.0 | 13.0 | 7.0 | 15.0 | 16.0 | 24.0 | 18.0 | 24.0 |
| Communication | 25.0 | 15.0 | 17.0 | 15.0 | 5.0 | 16.0 | 13.0 | 0.0 | 6.0 | 14.0 | 23.0 | 23.0 | 17.0 | 23.0 |
| Storage | 19.0 | 9.0 | 15.0 | 17.0 | 7.0 | 18.0 | 7.0 | 6.0 | 0.0 | 8.0 | 17.0 | 17.0 | 11.0 | 17.0 |
| Electrical | 21.0 | 17.0 | 23.0 | 25.0 | 15.0 | 26.0 | 15.0 | 14.0 | 8.0 | 0.0 | 25.0 | 15.0 | 9.0 | 15.0 |
| MedBay | 8.0 | 8.0 | 14.0 | 22.0 | 24.0 | 19.0 | 16.0 | 23.0 | 17.0 | 25.0 | 0.0 | 14.0 | 20.0 | 14.0 |
| Security | 6.0 | 16.0 | 22.0 | 30.0 | 24.0 | 27.0 | 24.0 | 23.0 | 17.0 | 15.0 | 14.0 | 0.0 | 6.0 | 5.0 |
| Lower E | 12.0 | 20.0 | 26.0 | 28.0 | 18.0 | 29.0 | 18.0 | 17.0 | 11.0 | 9.0 | 20.0 | 6.0 | 0.0 | 6.0 |
| Reactor | 6.0 | 16.0 | 22.0 | 30.0 | 24.0 | 27.0 | 24.0 | 23.0 | 17.0 | 15.0 | 14.0 | 5.0 | 6.0 | 0.0 |

Figure 4: Floyd Warshall Matrix for Crew mates

# 5 Step 4 : Secure the last task